Fenix - Personalized Information Filtering System for WWW Pages

Flávia Coimbra Delicato[o], Luci Pirmez, Luiz F. Rust da Costa Carmo

Núcleo de Computação Eletrônica - Universidade Federal do Rio de janeiro Tel: 21 5983159
Caixa Postal: 2324 Rio de Janeiro RJ Brasil
e-mails: flavia@eng.uerj.br, luci, rust@nce.ufrj.br
[o] Master Student in Informatic at the Institute of Mathematics of UFRJ

Abstract

Nowadays Internet offers a large amount of information to a wide range of users, making it difficult to deal with. The present work suggests the use of intelligent agents for the personalized filtering of Web pages. A set of autonomous, non-mobile and adaptive agents was developed aiming to satisfy the user's need for information. The agents learn from the users' feedback and attempt to produce better results over time. This work presents the system description and the promising results of tests performed in a simulated environment. The proposed system has proven to be a useful tool in reducing the amount of information the user has to deal with.

Keywords

Personalized Information Filtering, Web Pages, Agents.

1. Introduction

The exponential increase in the amount of information available on the Internet has been causing a significant impact on users. The World Wide Web (WWW) has made the Internet accessible to a wide range of users. Although the increase of the information available facilitates the spreading of knowledge and the acquisition of products and services, it also makes the search for relevant material a real challenge. Questions arise as how the users will be able to locate the information they need or how they will find the best offer for a service. The use of agents is a possible solution to this problem.

Agents can be defined as software with the aim of performing tasks for their users, usually with autonomy, playing the role of personal assistants.

The present work suggests the use of intelligent agents for personalized information filtering. The proposed system is composed of a set of autonomous, adaptive and non-mobile agents aiming to satisfy the user's need for information. The agents receive the user's feedback about the relevance of the retrieved information and improve their search, obtaining better results over time.

The set of agents is autonomous as it can perform its task without the user's presence, based on a previously built preference profile. The system is adaptive as it learns the user's preferences and adapts itself when these change over time. The agent's learning mechanisms are relevance feedback (Rocchio, 1971) and genetic algorithms (Goldberg, 1994). The information is represented by the vector space model (Salton, 1989), where queries and documents are represented as vectors in a vector space. This method was chosen for its

efficiency, which has been proven in various works in the area of information retrieval (Sheth, 1994) (Balabanovic, 1997) and for its relatively easy implementation.
The results presented were obtained through a series of sessions with simulated users. The system's efficiency evaluation was made through the normalized distance performance measure (ndpm), suggested by Yao (1995).
This paper is organized as follows: In section 2 there is a comparison with related works. Section 3 describes the system, the development methodology and environment. The system architecture is detailed in section 4. The analysis of results is presented in section 5 and, finally, some conclusions are drawn in section 6.

2. Comparison with Previous Works

Feedback relevance techniques have been studied in the context of information filtering tasks, as described at the TREC Conferences (Harman, 1994). There are also many comparisons between these techniques and non-incremental Machine Learning techniques (Schutze *et al*, 1995) (Lang, 1995). One of the disadvantages of the non-incremental techniques is the great number of examples needed before the learning algorithms can be applied. In contrast, the present work adopts a model where the user is presented to pages that are gradually better, where influences on the pages from his feedback will be presented. . The Newst system (Sheth, 1994) is a software agent which uses relevance feedback and genetic algorithms (GA) to provide personalized filtering of Usenets news. The approach differs from the present work mainly in the search space. When considering the filtering of WWW pages instead of news, there is a great impact on the information representation model and on the heuristics adopted in the GA. InfoScope (Fisher, 1991) learns by using systems based on rules that register interesting topics covered in the past. Recommendations of new topics are made based on how recent, frequent and spaced these past topics are. The disadvantage of such an approach is that it is restricted to recommendations of topics within the domain of the user's past interests. In contrast, in the proposed system, the agents search new domains for information that can be of potential interest to the user. He has probably never seen the presented topic before.
Balabanovic (1998) proposed a multiagent system which combines both content-based and collaborative techniques applied to the web pages recommendation. That work adopts the vector-space model (Salton, 1989), relevance feedback (Rocchio, 1971) as the learning method basis and he suggests the use of genetic algorithms (Goldberg, 1994) as a possible solution to some of the problems found in the content-based filtering.

3. Fenix System

The information filtering task involves repeated interactions over multiple sessions with the users having long-term goals. It differs from the information retrieval systems where the users typically have a short term information need that is satisfied in a single session. Information filtering systems assist users by filtering the data stream and delivering the relevant information to the user. Information preferences vary greatly from one user to another, therefore filtering systems must be highly personalized.
In the information filtering system proposed, an agent is modeled as a set population of individual profiles. As a whole, all the profiles in a set try to satisfy the user's interests and adapt themselves to these interests.

The agent is responsible for starting the execution of search and filtering tasks, one for each profile. As the tasks are autonomous, they are sub-agents in the Fenix system. Each sub-agent, using different search engines, goes through the web pages looking for documents containing the keywords provided by the user. The set of documents obtained undergoes the filtering process, according to the adopted model. The selected documents are the ones with the higher degree of similarity with the respective profile. The agent has to gather the results from all the sub-agents, classify them according to their potential relevance, and present them to the user. The user can provide positive or negative feedback for the documents. User feedback has the effect of modifying the profile used to retrieve that document.

3.1 Development Methodology and Environment

The Fenix system was developed according to the object oriented approach. The programming language adopted was Java, by Sun Microsystems Inc., and the development environment was Jbuilder Standard 1.0, by Borland Corporation, that uses JDK (Java Development Kit) version 1.1. Java language allows the implementation of two kinds of programs: applets and applications. The applications run on any platform with Java Virtual Machine (JVM), while applets are special programs designed to be executed from HTML pages. Fenix system was implemented as a Java application to be run locally.

4. Architecture

The Fenix system is composed of five functional modules (figure 1). The modules are implemented as groups of related classes. The description of each module is given below.

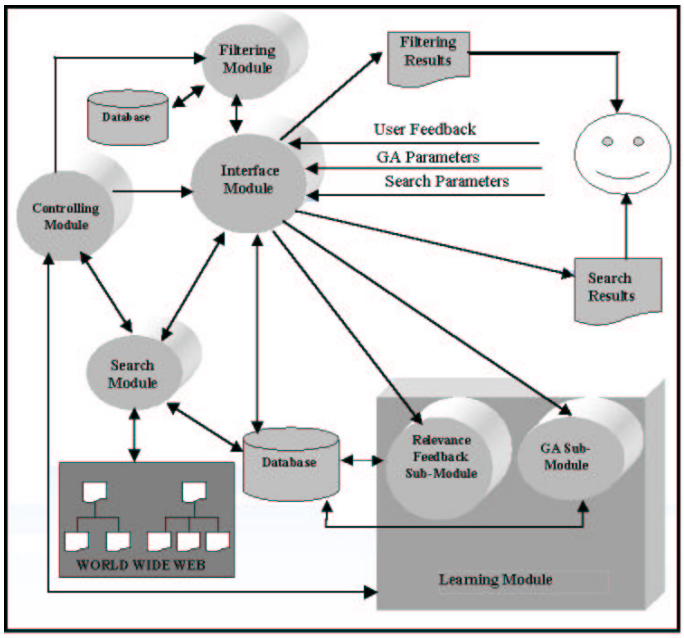

Figure 1: System's architecture

4.1 User Interface Module

This module presents a graphic interface to interact with the user. The user's interaction with Fenix system begins with his registration, where he must inform his personal data and choose

a login and a password. After identification the user can choose from three options: to create a new agent, to load an existing one or to activate the autonomous mode.
When creating a new agent, the user must choose a name and a background color, and provide the following search parameters: maximum number of documents shown per session (default is 30); and the query expression. A query in the Fenix system is a combination of keywords (technically called terms), separated by blank spaces. The use of logical connectives is not allowed. The presence of the connective AND between the terms is automatically assumed.
As a result of the initial search, a series of retrieved documents is presented. After reading the chosen documents, the user can provide positive (+1) or negative (-1) feedback according to their relevance.
When saving a newly created agent, the references to the documents with positive feedback will be saved (their URLs) and the term vector and their weights will be created, building the initial profiles for that agent.
When loading an existing agent, the user can choose one of three actions: to read some retrieved document, to provide feedback about some document or to start a new search for documents.
The Fenix "Autonomous Mode" works performing search and filtering tasks based on the user's profiles without his interaction.

4.2 Search Module

This module is responsible for gathering information from web pages about the chosen subject and saving them in a local database. With the development of this work, there was a choice of using existing search engines, such as Altavista[1], Lycos[2] and others. Then, the search module became responsible for making the interface with these mechanisms.
The system keeps a list of search engines as part of its configuration parameters. The user's keywords are provided to those engines and the first page of results retrieved for each one of them are converted to the string form and processed by the search module. The processing includes going through the pages identifying all the significant links, extracting the textual content of the pages pointed by the links and delivering this content to the filtering module.

4.3 Filtering Module

The filtering process consists of translating the documents provided by the search module to their vector representations, calculating the similarity between documents and profiles, and selecting the top-scoring documents for presentation to the user.

4.3.1 Profiles and Documents Representation

The representation adopted in this work is based on the vector space model (VSM) (Salton, 1989). According to it, documents and queries are represented as vectors in a hyper-space. A metric distance, which measures the proximity between vectors, is defined over space. The filtering results are the documents with representation that have the highest degree of proximity to the query vector.
A standard method for indexing texts consist of removing punctuation marks, recognizing individual words, eliminating functional words (as "and", "that", etc) using a stop-list, and

[1] Available in: http://www.altavista.digital.com
[2] available in: http://www.lycos.com

using the remaining words for content identification of the text. Since the terms are not equally important for content representation, weights are assigned to them, in proportion to their presumed importance to content identification purposes. A text is then represented as a vector of terms $T_i = \{W_{ij}\}$, where W_{ij} represents the weight of term t_j in text T_i.
In the representation adopted, the term weight is the product of the term frequency and the inverse document frequency. The term frequency (tf) is the occurrence frequency of the term in the text and it usually reflects the relevance of this term. The inverse document frequency (idf) is a factor that enhances the terms that appear in few documents, while it devaluates the terms occurring in many documents. As a result, the specific features of the documents are highlighted, while the ones spread through the set of documents have minor importance. The weight of the terms is then given as:

$$W_{ij} = tf_{ik} \times idf_k , \quad (1)$$

where tf_{ik} is the number of occurrences of term t_k in document i, and idf_k is the inverse document frequency of term t_k in the collection of documents. A commonly used measure for idf is $idf_k = \log (N/nk)$, where N is the total number of documents in the collection, from which nk contains a term t_k. In this work, a collection of documents is formed by all the documents retrieved by a profile.
A profile is a set of information about the retrieved documents as, for example, their location in the net (URL) and the user's feedback assigned to them. Besides this, it contains the vector representation of documents that got positive feedback. The representation consists of a vector of terms similar to the one previously described for documents:

$$P = \{W_{ij}^{p}\}, \quad (2)$$

where "p" indicates a profile-field other than a document field.

4.3.2 Evaluation of Filtered Documents

A commonly used similarity measure in the vector space model is the cosine of the angle between vectors. In the proposed application, different formulae were tested, and the one proposed in (Salton, 1989) was adopted.

4.3.3 Scoring and Selecting Documents

The documents retrieved through the search task started by the respective profile will have their similarities calculated in relation to that profile. The agents are responsible for gathering the documents generated by all the profiles, classifying them according to their similarity values, eliminating repetitions and presenting them to the user.
The similarity values are converted to a class scale to be presented to the user. A five points scale was adopted, with the adjectives:

- Terrible: for scores equal to 0.2
- Poor: for scores between 0.2 and 0.3
- Neutral: for scores between 0.3 and 0.5
- Good: for scores between 0.5 and 0.8
- Excellent: for scores greater than 0.8

The maximum score value is 1.0 and it only happens when the profile and the document representations are identical. In this work, the limit of 0.2 was adopted as the minimum score that a document should have in order to be presented to the user.

4.4 Learning Module

The learning methods adopted by the system are relevance feedback and genetic algorithms. Both methods were projected as independent sub-modules and only the relevance feedback has been implemented at the present stage. In the next stage, the genetic algorithm will be implemented as a complementary mechanism, introducing diversity to the search parameters used by the agents as a goal.

4.4.1 Relevance Feedback Sub-Module

The user's feedback for a document has the effect of modifying the respective profile. The profile has to incorporate the changes before new documents can be evaluated.
In the relevance feedback method, an original query vector (represented by the profiles) is modified based on the user's feedback for the documents retrieved by the profile.
For vector space representations, the method for query reformulation in response to user's feedback is vector adjustment. Since queries and documents are both vectors, the query vector is moved closer to the vector representing documents with positive feedback, and further from the vectors of the documents with negative feedback. This effects the weight of the terms already existing in the profile by modifying them in proportion to the feedback. The terms not existing in the profile must be added to it.

4.5 Other Modules

The Fenix system database is composed of all information from the user, his agents and respective profiles, as well as the documents retrieved in searches. This database is implemented through a group of classes existent in Java standard APIs. Users' and agents' data are stored in simple text files. Profiles and documents are stored in object files.
Besides all of these modules, Fenix has a controlling module (figure 1) responsible for controlling the global behavior of the system, the behavior of each user's set of agents and the specific behavior of each individual agent.

5. Analysis of Results

In this work, the performance measure proposed by Yao was adopted (Yao, 1995). The ndpm measure ("normalized distance-based performance measure") is a distance, normalized to range from 0 to 1, between the user's classification for a set of documents and the system's classification for the same documents. This will provide a relative measure, that will be more appropriate to the system's goal than recall and precision measures, commonly used in information retrieval.
An outline was adopted as suggested in (Balabanovic, 1997). A special list of documents is supplied to a simulated user who should classify it in agreement with his interests in a subject. That list is randomly selected from several documents retrieved from the web. The system also ranks the documents according to how well they match the profile previously built for that user. The expected result is for the ndpm distance between the user and system classifications to decrease gradually over time, as the user's profile is adjusted.
One hundred agents were created for thirty subjects of interest of a simulated user. For each agent, six simulated sessions of "user"-agent interaction were performed. After an initial search, the agent classified the retrieved documents according to the categories above, the "user" evaluated the documents, providing their feedback values and classification. With the

feedback, the agent's profile was adjusted to later searches and, with the classification, the ndpm distance was calculated.
A progressive decrease of that distance in the course of the sessions (Figure 2) was observed, indicating that the agents were adapting themselves to the user's preferences, increasing, then, the probability of retrieving a larger number of relevant documents and discarding the irrelevant ones.
The obtained results are similar to the ones described in (Balabanovic, 1998), where a multiagent system was implemented for the WWW pages recommendation. Balabanovic proposed an architecture that combined content-based with collaborative filtering. His system performance was also evaluated with the ndpm measure and the obtained curve had a behavior quite similar to the one presented in the tests with Fenix (Figure 2). In his work 25 evaluation sessions were performed. The initial average ndpm values were 0.4 and the final values were 0.001.
Several system configuration parameters were tested in the simulated sessions. To sum up the final results we can say that the system reaches the best performance when the query expression is more specific, the agents are composed by at least 4 (for) and in the maximum 10 (ten) profiles and the term vectors of the documents have maximum size of 300 terms.

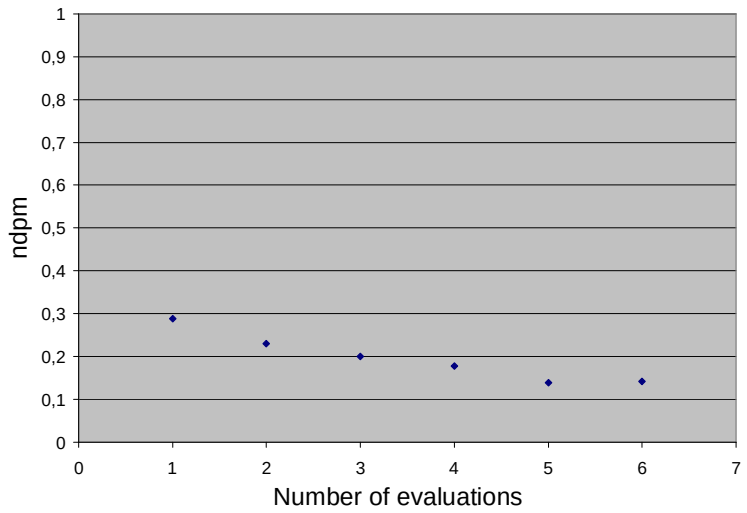

Figure 2: Average ndpm distance between user and system rankings, over all agents at evaluation points.

6. Conclusions

An information filtering system must be capable of becoming specialized according to user's interests, getting adapted to changes and exploring the domain of potentially relevant information.
The results of simulated tests using the ndpm distance in a controlled environment were quite satisfactory. The tests showed that the distance between the evaluation provided by the user and the one provided by the agents decreased in the course of the sessions, indicating the agents' adaptation to the user's interests. These results demonstrate that the relevance feedback

technique alone is very efficient for the agent to become specialized in the user's specific interests.
The performance values obtained in simulated tests based on the ndpm measure were similar to the ones found in other works of information filtering. Balabanovic (1998) proposed an approach that combines content-based and collaborative techniques applied to the WWW pages recommendation. The results described in his work were quite similar to the obtained with Fenix, where a simpler architecture was adopted.
A question that has arisen in the tests was the low scalability of the system. As a future work, a possible solution for that problem is the implementation of a system with multiple search agents collecting pages Web in several sources. The search agents would deposit the pages obtained in a central repository where individual users' personal agents would recover those that best satisfied the users' profiles. The supplied feedback would go both to the personal agent (that would adjust the user's profile) and to the search agents, that would be serving the users' groups instead of individual users and taking advantage of the shared interests.
The impact of systems like Fenix on an enterprise may be quite significant. Several information search tasks could be automated, as the system is able to filter and classify different subjects of interest, therefore reducing the consumption of time and money spent on those activities.
Information filtering agents are a great promise to the management of extensive available information.

7. References

Balabanovic, M. (1997) “An Adaptive Web Page Recommendation Service”. *Stanford Universal Digital Libraries Project Working Papers SIDL - WP.*

Balabanovic, M.(1998) “Learning to Surf: Multiagent Systems for Adaptive Web Page Recomendation Service”. Dissertation submitted to the Department of Computer Science and the Committee on Graduate Studies of Stanford University. UMI Number: 9837173. UMI Company.

Fischer, G.and Stevens, C. (1991), “Information access in complex, poorly structured Information spaces”. *Human Factors in Computing Systems CHI'91 Conference Proceedings*, New Orleans, LA, USA. April 27 - May 2, 1991, pp. 63-70.

Goldberg, D. E. (1994) “Genetic and Evolutionary Algorithms come of age”. *Communications of the ACM*, 37(3):113-119, March 1994.

Harman, D. (1994). *Overview of the Third Text Retrieval Conference (TREC-3).* In Proceedings of the 3rd Text Retrieval Conference. Gaithersburg, Maryland, USA. November, 2 - 4, 1994.

Lang, K. (1995). “NewsWeeder: Learning to filter netnews”. In *Proceedings of the 12th International Conference on Machine Learning.*

Rocchio, J.J. (1971) “Relevance feedback in information retrieval”**.** In *The Smart Retrieval System - Experiments in automatic Document Processing*, p. 313-323, Englewood Cliffs: Prentice-Hall.

Salton, G.(1989), *Automatic Text Processing – The Transformation, Analysis and Retrieval of Information by Computer*. Addison-Wesley Publishing Company, Inc., Reading, MA.

Schutze, H.,Hull, D and Pedersen, J. O. (1995) “A comparison of classifiers and documents representations for the routing problem”. In *Proceedings of the 18 th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Seattle, WA, USA. July 9 - 13.

Sheth, B. (1994) “NEWT: A learning approach to personalized information filtering”.Thesis.[s.l.:1994]. Available in: http://agents.www.media.mit.edu/groups/agents/ papers/newt-thesis/ tableofcontents2_1.html.

Yao, Y. Y. (1995). “Measuring retrieval effectiveness based on user preference of documents”. *Journal of the American Society for Information Science* 46(2):133-145.